

Vitis-Tutorials/Hardware_Acceleration
/Feature_Tutorials
/03-dataflow_debug_and_optimization

612k0023c

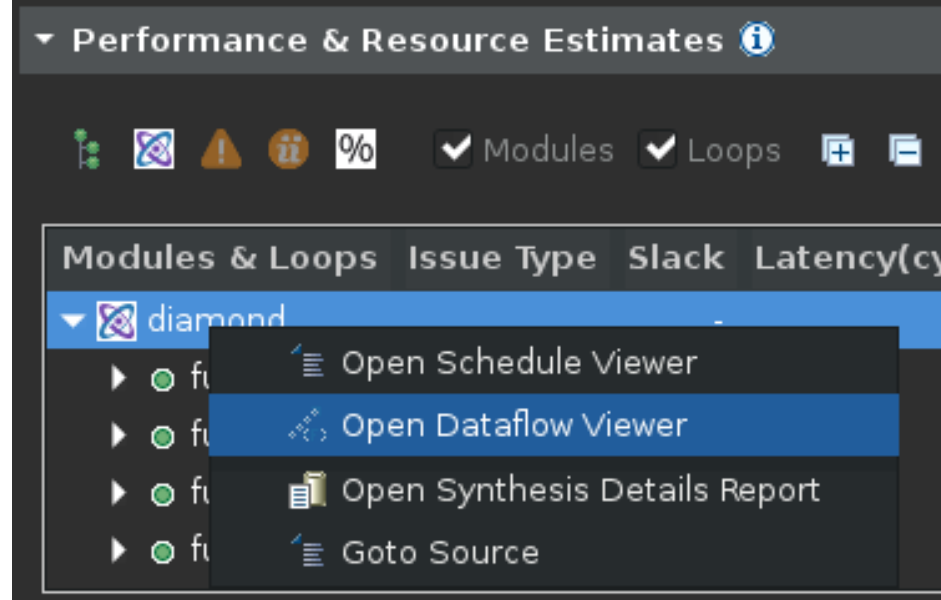
Hao Chen

Outline

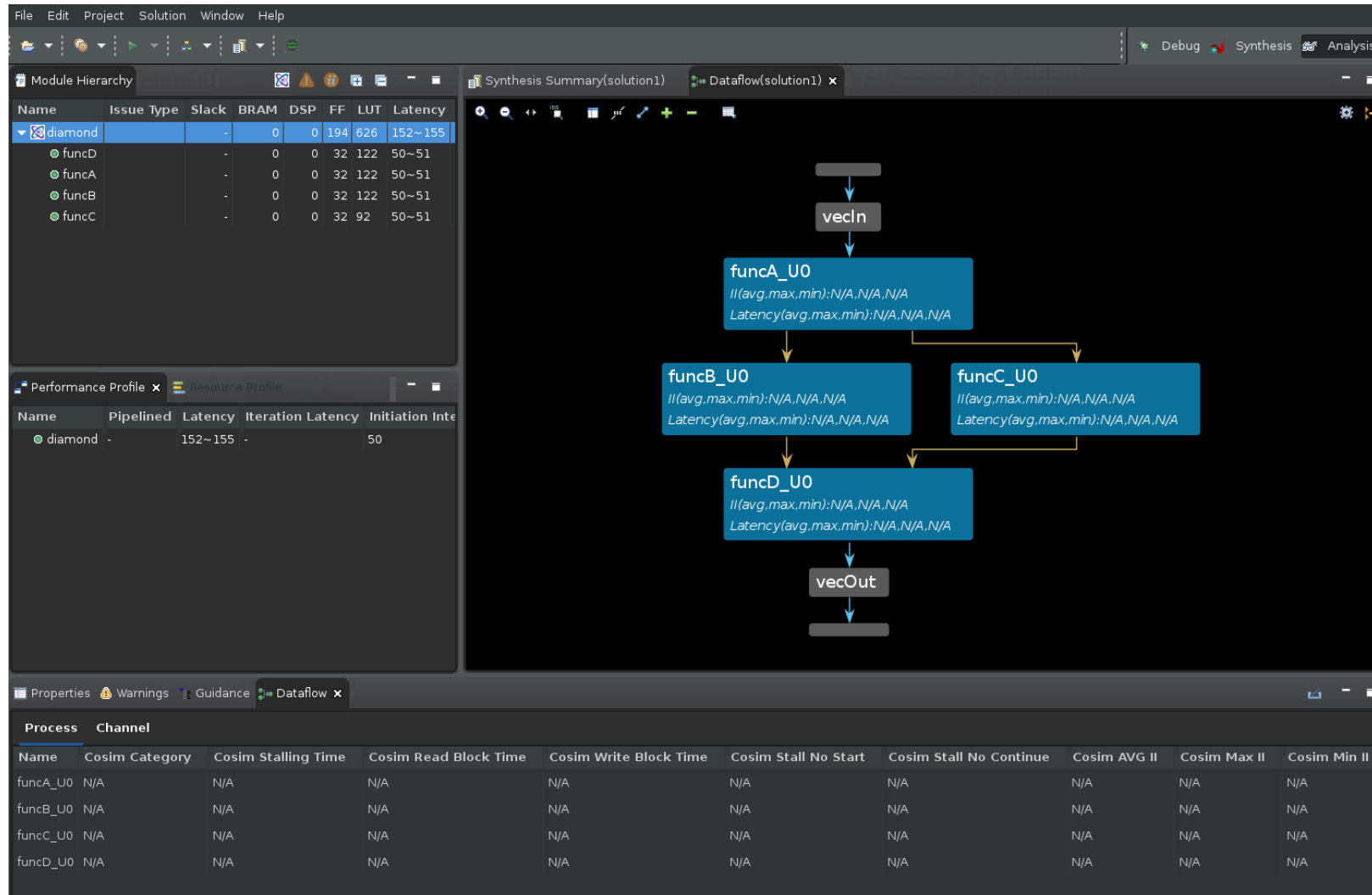
- Dataflow Viewer Basics
- FIFO Sizing and Deadlock

First Lab – Dataflow Viewer basics

- `$cd 03-dataflow_debug_and_optimization/reference-files/dataflow`
- `$vitis_hls -p script.tcl`
- **Run C Simulation / C Synthesis**
- **Open Dataflow Viewer:**

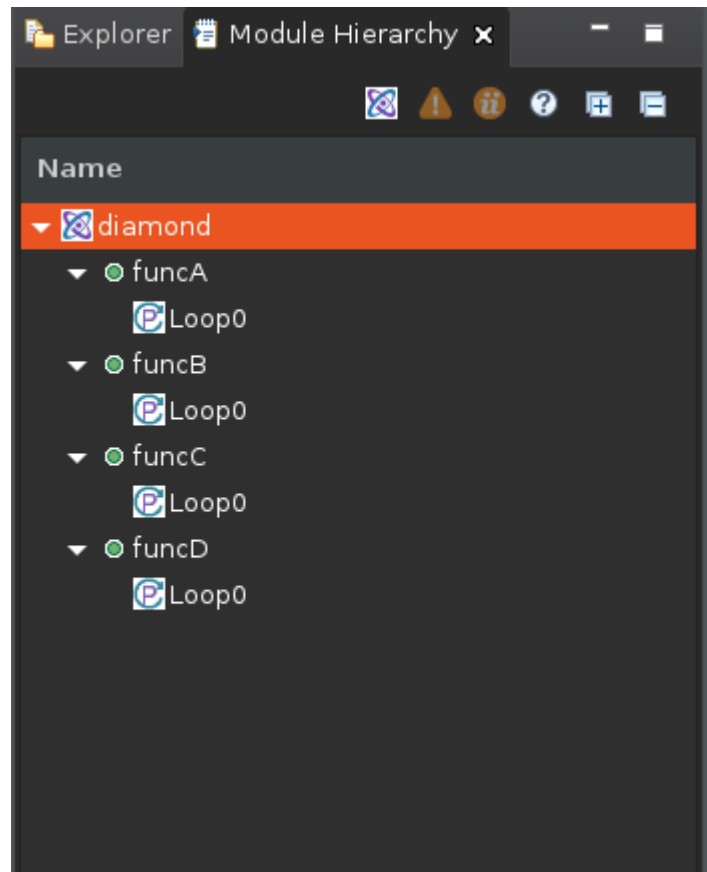


First Lab – Dataflow Viewer basics Cont.

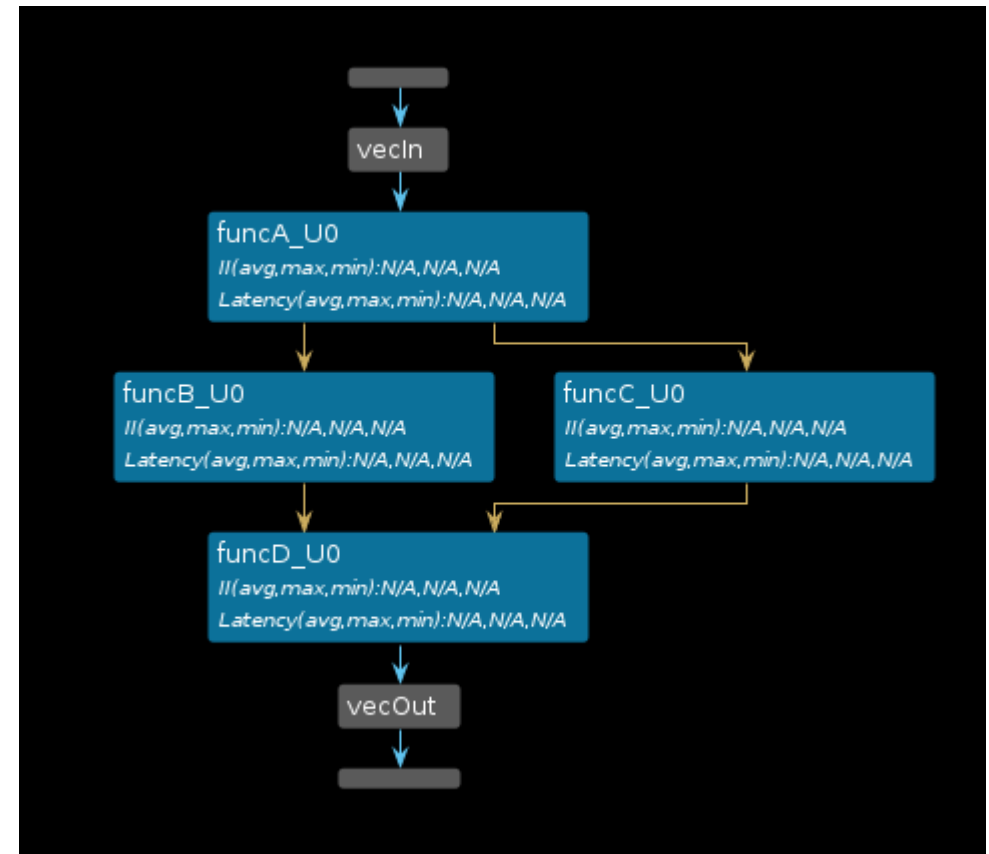


First Lab – Dataflow Viewer basics Cont.

- Module Hierarchy view:

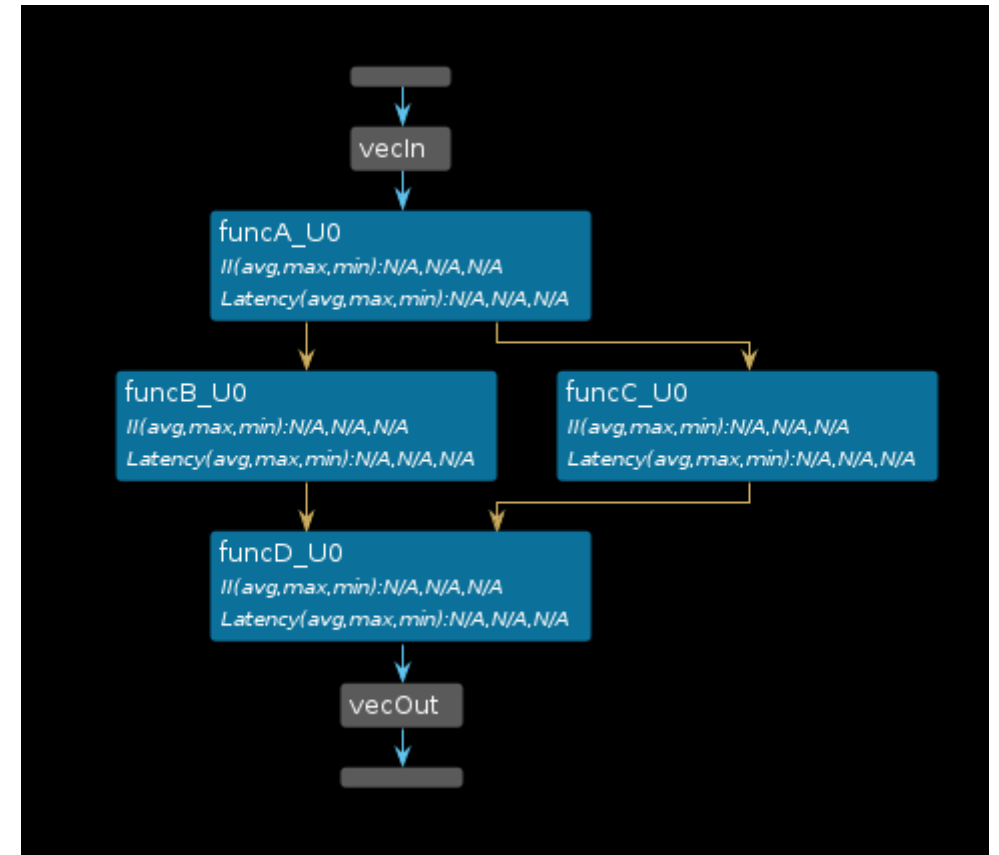


- The Dataflow Graph Pane



First Lab – Dataflow Viewer basics Cont.

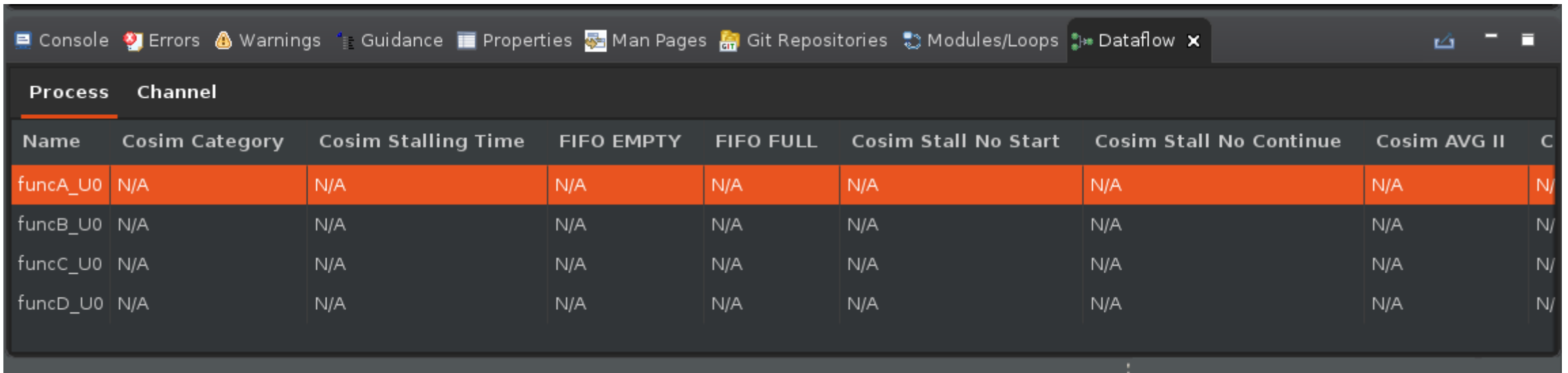
- The Dataflow Graph Pane
 - The edges in the graph (shown as blue, green, or gold arrows) represent the communication between the functions:
 - Blue edges represent data dependencies between the functions
 - Green edges represent the inferred FIFO channels between the functions
 - Gold edges represent the inferred PIPO channels



First Lab – Dataflow Viewer basics Cont.

- Dataflow Properties Table

- At the bottom part of the GUI is the Dataflow Properties table which shows various details about the dataflow processes (or functions) and the dataflow channels.



The screenshot shows the Dataflow Viewer GUI with a dark theme. At the top, there is a toolbar with icons for Console, Errors, Warnings, Guidance, Properties, Man Pages, Git Repositories, Modules/Loops, and Dataflow. The Dataflow tab is active, displaying a table with the following columns: Process, Channel, Name, Cosim Category, Cosim Stalling Time, FIFO EMPTY, FIFO FULL, Cosim Stall No Start, Cosim Stall No Continue, Cosim AVG II, and C. The table contains four rows of data, all with 'N/A' values except for the 'Name' column.

Process	Channel	Name	Cosim Category	Cosim Stalling Time	FIFO EMPTY	FIFO FULL	Cosim Stall No Start	Cosim Stall No Continue	Cosim AVG II	C
		funcA_U0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
		funcB_U0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
		funcC_U0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
		funcD_U0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

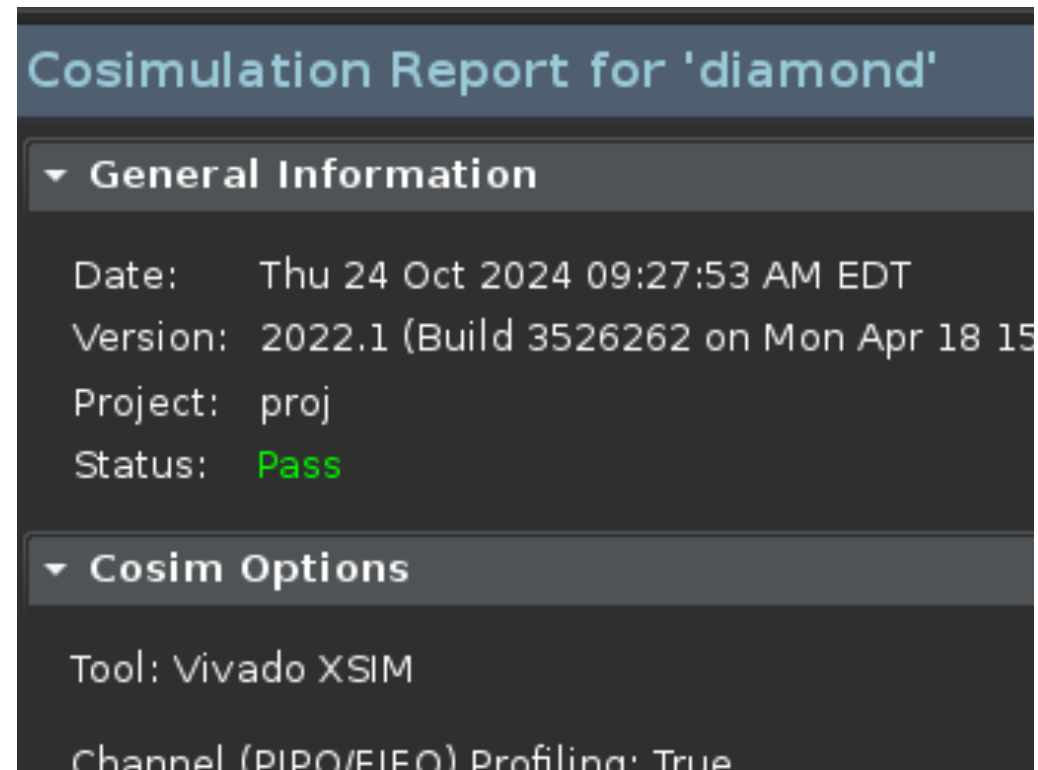
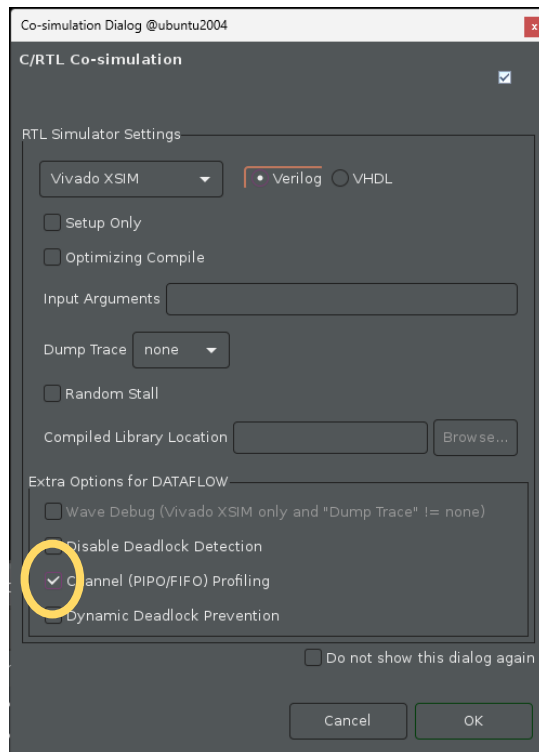
First Lab – Dataflow Viewer basics Cont.

- It is important to understand that the dataflow optimization is a dynamic optimization, unlike pipelining which is a static optimization. Because of this, while the compiler informs you of implementing the dataflow optimization, the effects of the optimization **cannot be seen until after running RTL co-simulation**. The process or channel details are marked as N/A as shown in the figure below until RTL co-simulation has generated performance data.

First Lab – Dataflow Viewer basics Cont.

Viewing the Dataflow Graph after RTL co-simulation

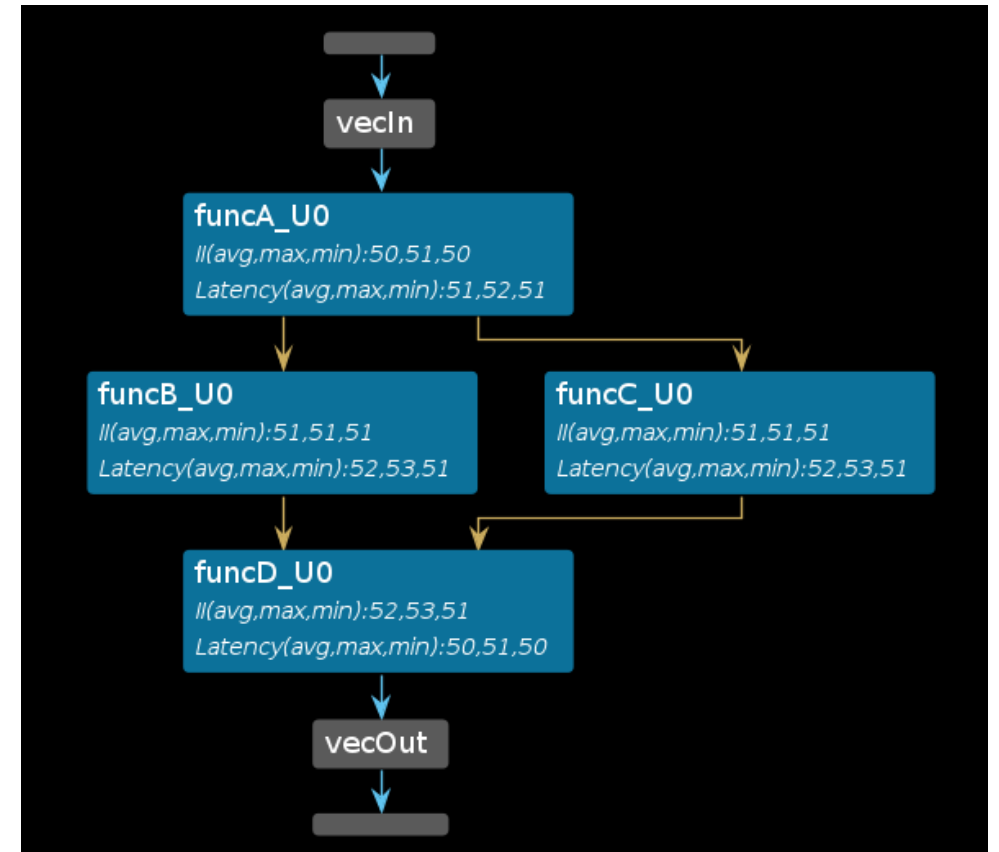
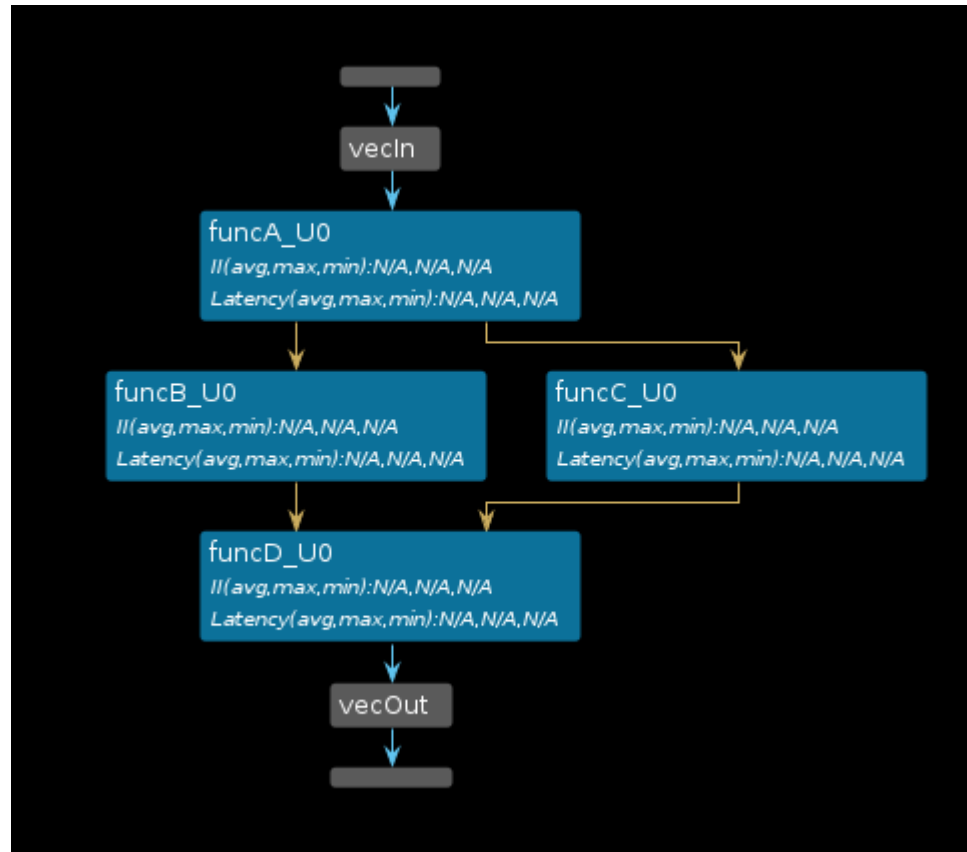
- select the **Solution > Run C/RTL Co-Simulation** command



First Lab – Dataflow Viewer basics Cont.

Viewing the Dataflow Graph after RTL co-simulation

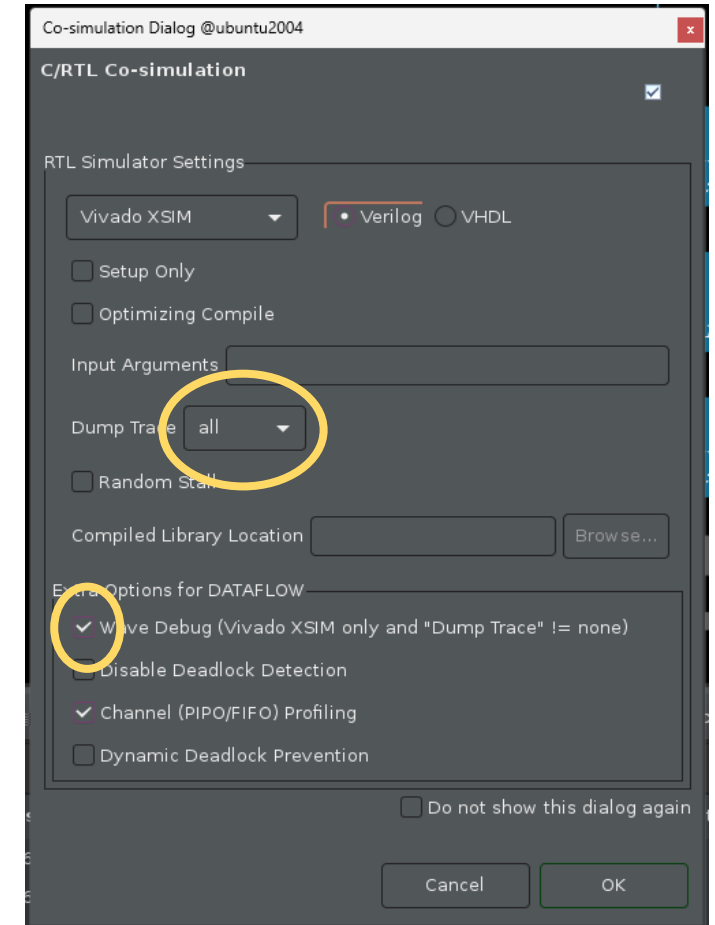
- After the co-simulation completes, select **Analysis** in the upper right hand corner of the screen to switch to the Analysis perspective.



First Lab – Dataflow Viewer basics Cont.

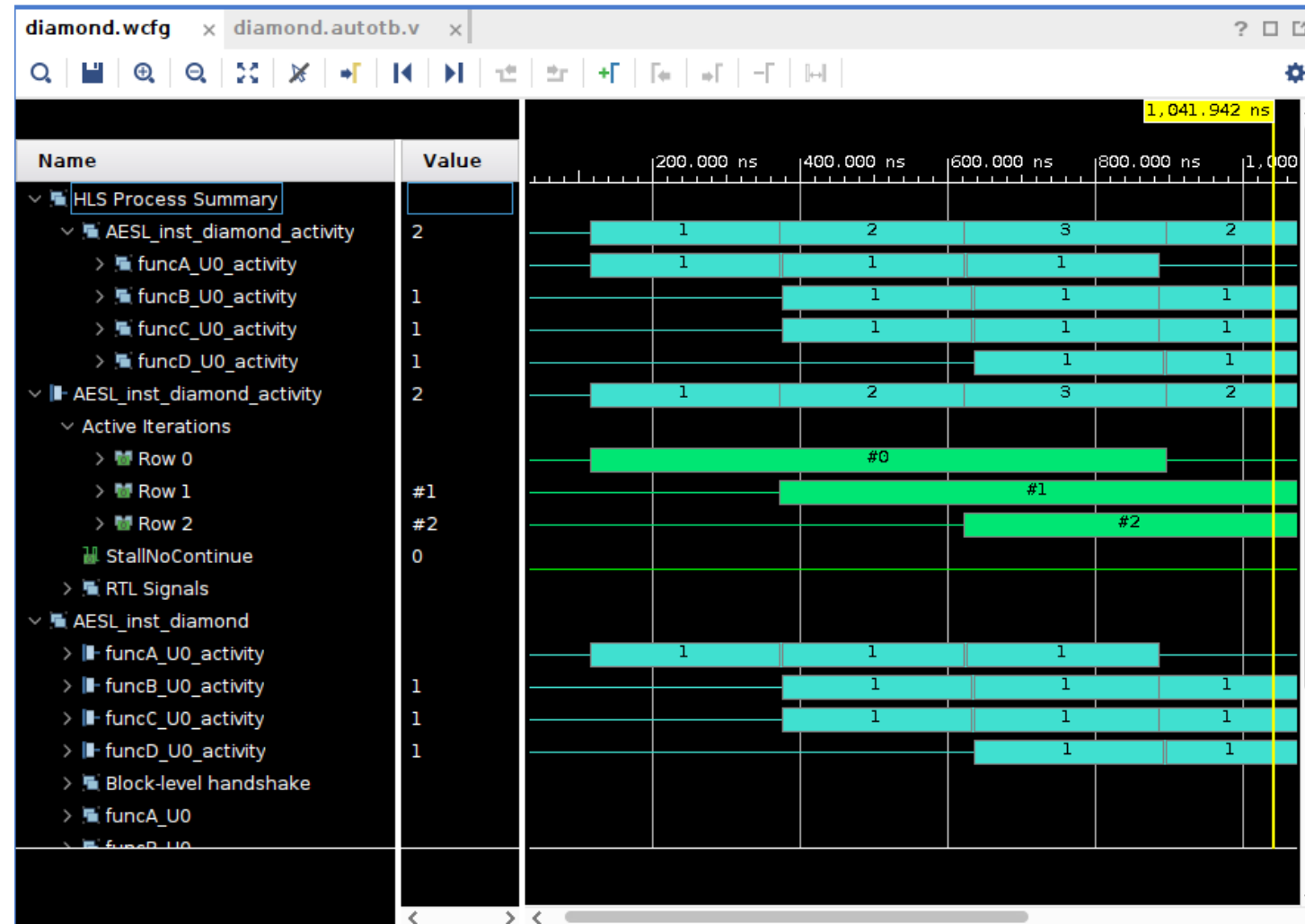
Viewing Dataflow Performance using Waveforms

- Ensure that the **Vivado XSIM** simulator is chosen
- Select **all** for the **Dump Trace** option to trace all ports and signals. Note: This is a small design and so we can dump and trace all the signals. For a large design, this might cause an increased simulation run time as well as the creation of a large waveform database.
- Enable the **Wave Debug** option to interactive launch the XSIM waveform viewer during simulation.
- Enable the **Channel (PIPO/FIFO) Profiling** checkbox.
- Click **OK**.



First Lab – Dataflow Viewer basics Cont.

Viewing Dataflow Performance using Waveforms



Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

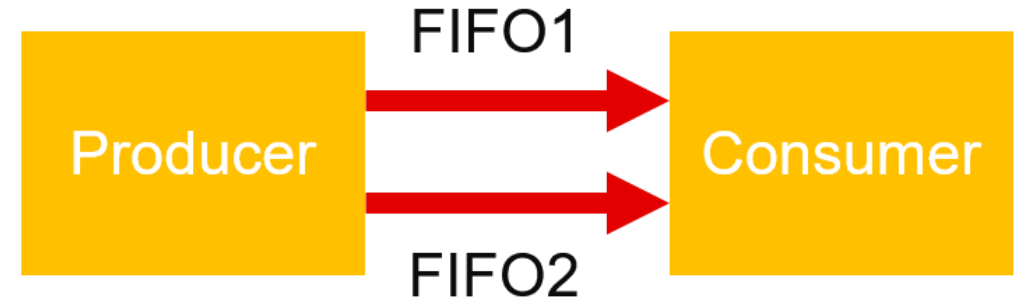
Types of channels

Channel Type	Examples	Created By
FIFO	Streams (including hls::streams and streamed arrays)	User
	Scalar propagation FIFOs	Tool
	Streams of blocks	User

Channel Type	Examples	Created By
PIPO	PIPO	User
	Task Level FIFOs (TLF)	Tool
	Input and output ports to the upper level	User

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Deadlock Detection and Analysis



- Case 1:

- Producer alternately writes to FIFO1, FIFO2, FIFO1, FIFO2, and so on.
- Consumer alternately reads from FIFO1, FIFO2, FIFO1, FIFO2, and so on.
- A depth of 1 for both FIFOs is enough to avoid deadlocks (and the default depth of 2 optimizes for performance).

- Case 2 (same structure):

- Producer writes to FIFO1 for N times, then to FIFO2 for N times
- Consumer alternately reads from FIFO1, FIFO2, FIFO1, FIFO2, and so on.
- A depth of N is necessary for FIFO1 (and the default depth of 2 for FIFO2 is optimal for performance)

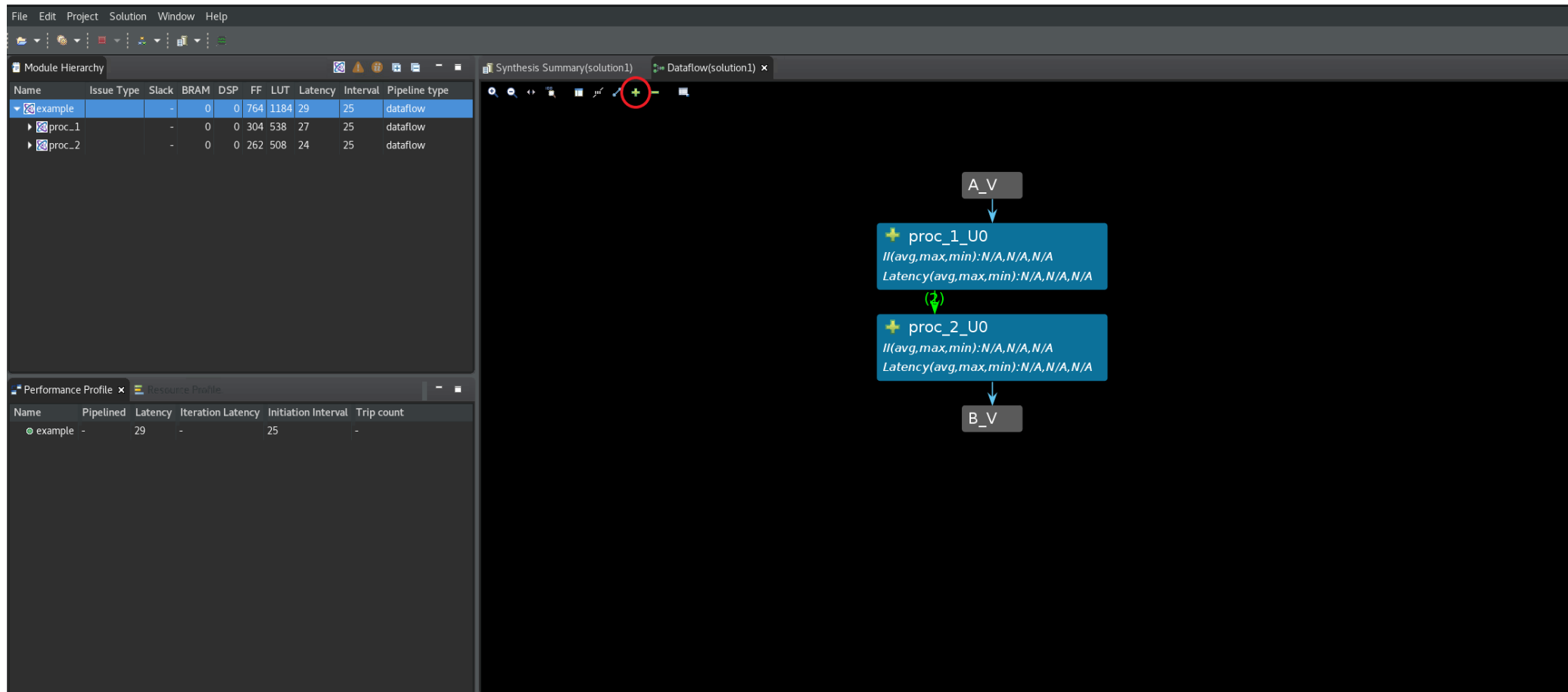
Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Deadlock Detection and Analysis

- `$cd 03-dataflow_debug_and_optimization/reference-files/deadlock`
- `$vitis_hls -p script.tcl`
- Run C-sim, C-synthesis, Co-sim

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Deadlock Detection and Analysis



Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Deadlock Detection and Analysis

The screenshot displays the Vivado IDE interface with the following components:

- Module Hierarchy:** A table listing modules and their properties.
- Performance Profile:** A table showing performance metrics for the 'example' module.
- Synthesis Summary:** A window showing the synthesis process.
- Dataflow Graph:** A visual representation of the dataflow, with a yellow box highlighting a deadlock state involving 'proc_1_1_U0' and 'proc_1_2_U0'.
- Properties:** A window showing properties for the 'Dataflow' module.
- Process Channel:** A table showing the process channel details, with a yellow box highlighting the 'Depth' column.

Name	Issue Type	Slack	BRAM	DSP	FF	LUT	Latency	Interval	Pipeline type
example		-	0	0	764	1184	25		dataflow
proc_1		-	0	0	304	538	27		dataflow
proc_2		-	0	0	262	508	24		dataflow

Name	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
example	-	29	-	25	-

Name	Cosim Category	Cosim Read Block Time	Cosim Write Block Time	Cosim Max Depth	Depth	Type	Sub-type	BitWidth	Producer	Consumer	Cosim Distribution Graph
data_channel1_read_block		60.00%	0.00%	0	2	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
data_channel2_read_block		60.00%	0.00%	0	2	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
data_channel1_write_block		0.00%	50.00%	2	3	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
data_channel2_read_block		60.00%	0.00%	0	2	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
data_channel1_read_block		50.00%	0.00%	0	2	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A
data_channel2_read_block		50.00%	0.00%	0	2	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A

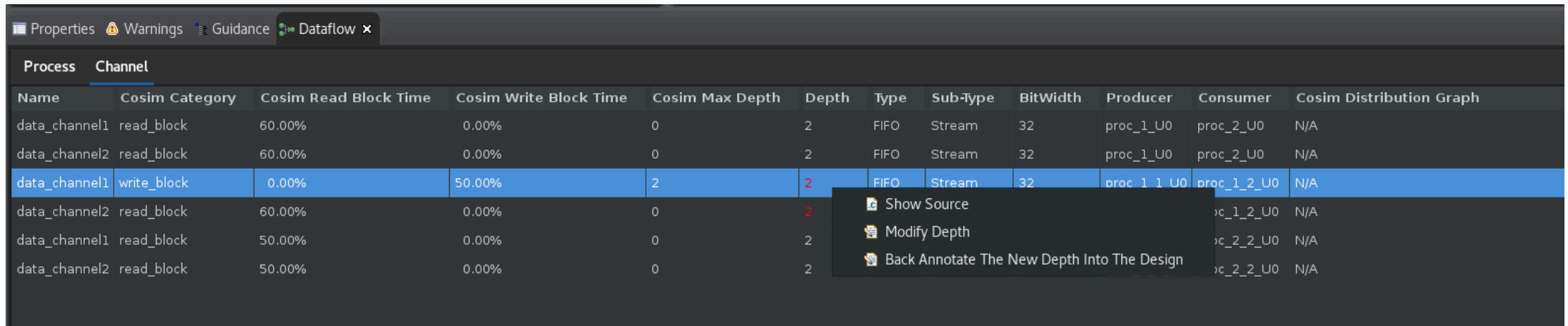
Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Deadlock Detection and Analysis

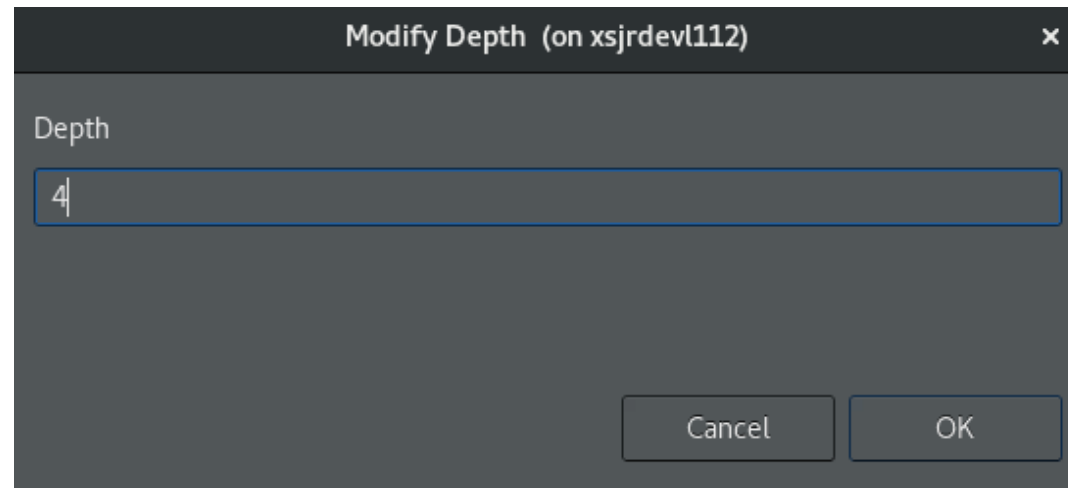
- There are three ways to do this FIFO sizing, and this lab will walk through each one in turn:
 - Manual FIFO sizing
 - Global FIFO sizing
 - Automated FIFO sizing

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Manual FIFO sizing



Process	Channel											
Name	Cosim Category	Cosim Read Block Time	Cosim Write Block Time	Cosim Max Depth	Depth	Type	Sub-Type	BitWidth	Producer	Consumer	Cosim Distribution Graph	
data_channel1	read_block	60.00%	0.00%	0	2	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A	
data_channel2	read_block	60.00%	0.00%	0	2	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A	
data_channel1	write_block	0.00%	50.00%	2	2	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A	
data_channel2	read_block	60.00%	0.00%	0	2					pc_1_2_U0	N/A	
data_channel1	read_block	50.00%	0.00%	0	2					pc_2_2_U0	N/A	
data_channel2	read_block	50.00%	0.00%	0	2					pc_2_2_U0	N/A	



Modify Depth (on xsjrdevl112) ✕

Depth

4

Cancel OK

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Manual FIFO sizing

Modify Depth (on xsjrdev112)

Depth: 4

Cancel OK

Module Hierarchy

Name	Issue Type	Slack	BRAM	DSP	FF	LUT	Latency	Interval	Pipeline type
example		-	0	0	764	1184	29	25	dataflow
proc_1		-	0	0	304	538	27	25	dataflow
proc_2		-	0	0	262	508	24	25	dataflow

Resource Profile

Latency	Iteration Latency	Initiation Interval	Trip count
29	-	25	-

Dataflow

Properties

Name	Cosim Category	Cosim Read Block Time	Cosim Write Block Time	Cosim Max Depth	Depth	Type	Sub-type	BitWidth	Producer	Consumer	Cosim Distribution Graph
data_channel1_read_block	66.67%	0.00%	0	0	2	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
data_channel2_read_block	66.67%	0.00%	0	0	2	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
data_channel1_write_block	0.00%	41.67%	4	4	4	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
data_channel2_read_block	66.67%	0.00%	0	0	2	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
data_channel1_read_block	58.33%	0.00%	0	0	2	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A
data_channel2_read_block	58.33%	0.00%	0	0	2	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Manual FIFO sizing

The screenshot displays the Vivado IDE interface during synthesis. The top panel shows the 'Module Hierarchy' with 'example' expanded to show 'proc_1' and 'proc_2'. The bottom panel shows the 'Properties' window with the 'Channel' tab selected, displaying a table of FIFO channels. The 'Depth' column in this table is circled in red, and red arrows point from these values to the corresponding FIFO blocks in the block design on the right.

Name	Issue Type	Stack	BRAM	DSP	FF	LUT	Latency	Interval	Pipeline type
example	-	-	0	0	764	1184	29	25	dataflow
proc_1	-	-	0	0	304	538	27	25	dataflow
proc_2	-	-	0	0	262	508	24	25	dataflow

Name	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
example	-	29	-	25	-

Name	Cosim Category	Cosim Read Block Time	Cosim Write Block Time	Cosim Max Depth	Depth	Type	Sub-Type	BitWidth	Producer	Consumer	Cosim Distribution Graph
data_channel1	read_block	36.11%	0.00%	10	29	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
data_channel2	read_block and write_block	69.44%	11.11%	2	2	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
data_channel1	none	0.00%	0.00%	10	10	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
data_channel2	read_block	33.33%	0.00%	1	10	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
data_channel1	read_block and write_block	69.44%	13.89%	2	2	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A
data_channel2	read_block	86.11%	0.00%	0	2	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Manual FIFO sizing

The screenshot displays the Vivado IDE interface for a project named 'example'. The top menu bar includes File, Edit, Project, Solution, Window, and Help. The main workspace is divided into several panels:

- Module Hierarchy:** A tree view showing the project structure. It includes 'example' (dataflow), 'proc_1' (dataflow), and 'proc_2' (dataflow).
- Performance Profile:** A table showing performance metrics for the 'example' module.
- Dataflow Graph:** A hierarchical block diagram showing the internal structure of the modules. It includes blocks like 'proc_1_U0', 'proc_1_1_U0', 'proc_1_2_U0', 'proc_2_U0', 'proc_2_1_U0', and 'proc_2_2_U0', connected by dataflow links.
- Properties Panel:** A table showing properties for the 'example' module, including Name, Channel, Cosim Read Block Time, Cosim Write Block Time, Cosim Max Depth, Depth, Type, Sub-type, BitWidth, Producer, Consumer, and Cosim Distribution Graph.

The Performance Profile table shows the following data:

Name	Pipelined	Latency	Iteration Latency	Initiation Interval	Trip count
example	-	29	-	25	-

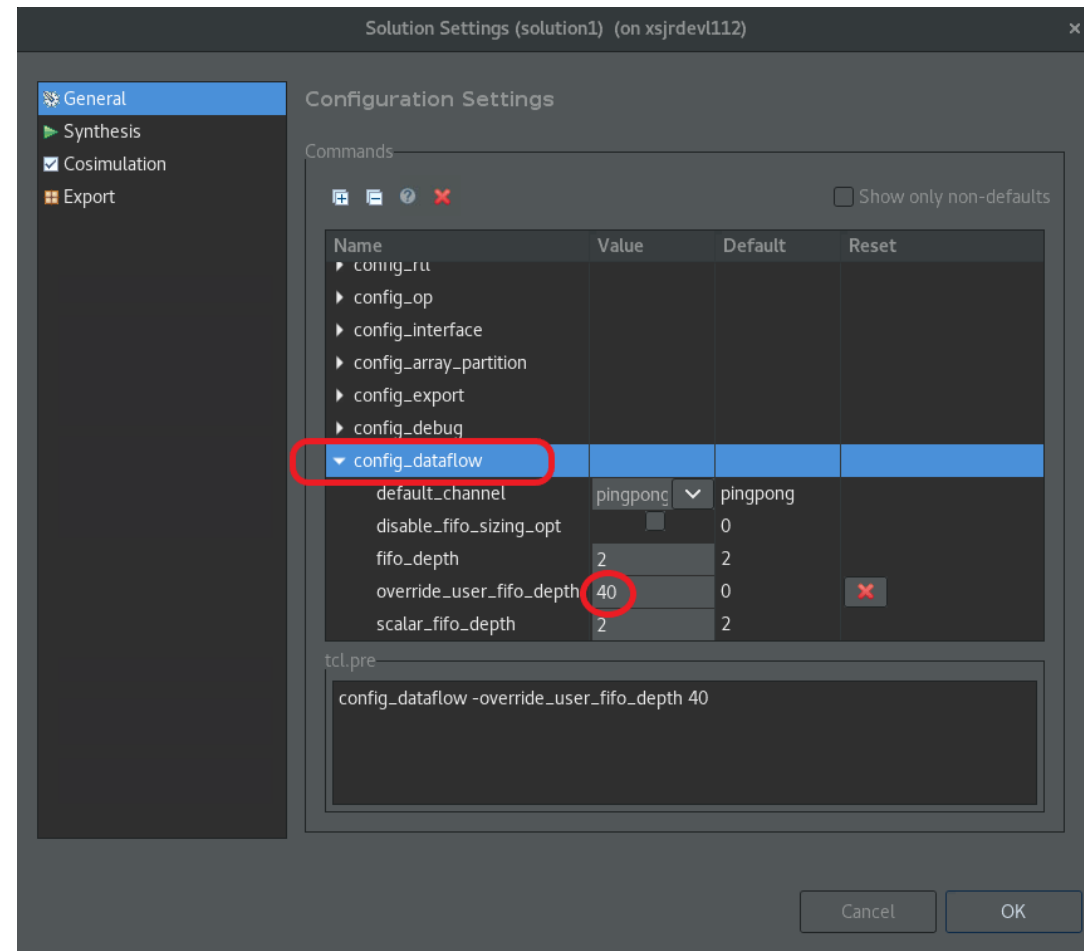
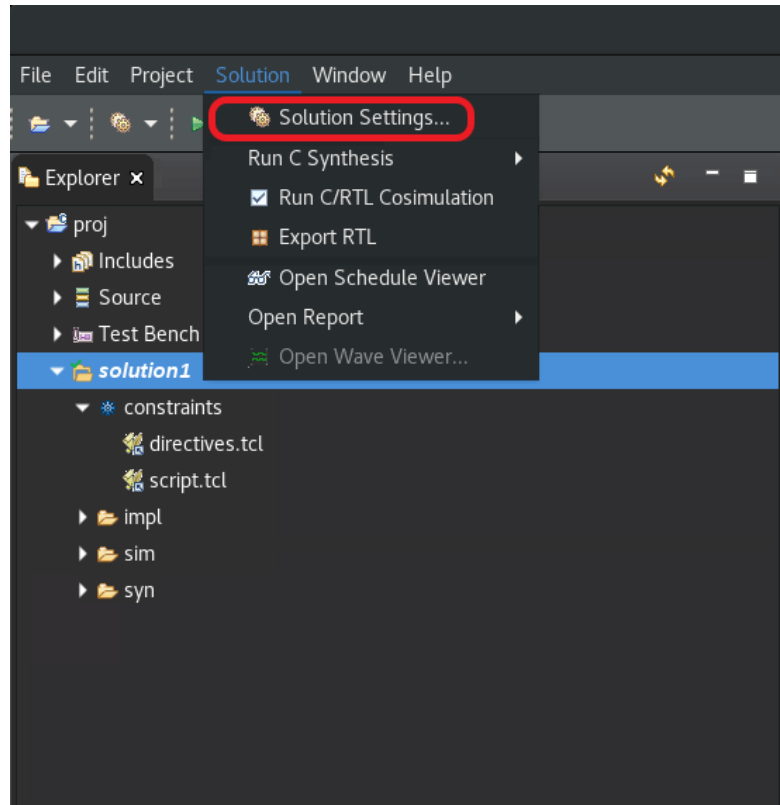
The Dataflow Graph shows a hierarchical structure of modules. The top-level module is 'example', which contains 'proc_1' and 'proc_2'. 'proc_1' contains 'proc_1_U0', which in turn contains 'proc_1_1_U0' and 'proc_1_2_U0'. 'proc_2' contains 'proc_2_U0', which contains 'proc_2_1_U0' and 'proc_2_2_U0'. The graph also shows dataflow links between these modules, such as 'A_V' and 'B_V'.

The Properties Panel shows the following data:

Name	Channel	Cosim Read Block Time	Cosim Write Block Time	Cosim Max Depth	Depth	Type	Sub-type	BitWidth	Producer	Consumer	Cosim Distribution Graph
data_channel1	read_block	10.00%	0.00%	10	10	FIFO	Stream	32	proc_1_U0	proc_2_U0	Link
data_channel2	read_block	20.00%	0.00%	1	10	FIFO	Stream	32	proc_1_U0	proc_2_U0	Link
data_channel3	none	0.00%	0.00%	10	10	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	Link
data_channel4	read_block	10.00%	0.00%	1	10	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	Link
data_channel5	read_block	20.00%	0.00%	10	10	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	Link
data_channel6	read_block	56.92%	0.00%	1	10	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	Link

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Global FIFO sizing



Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

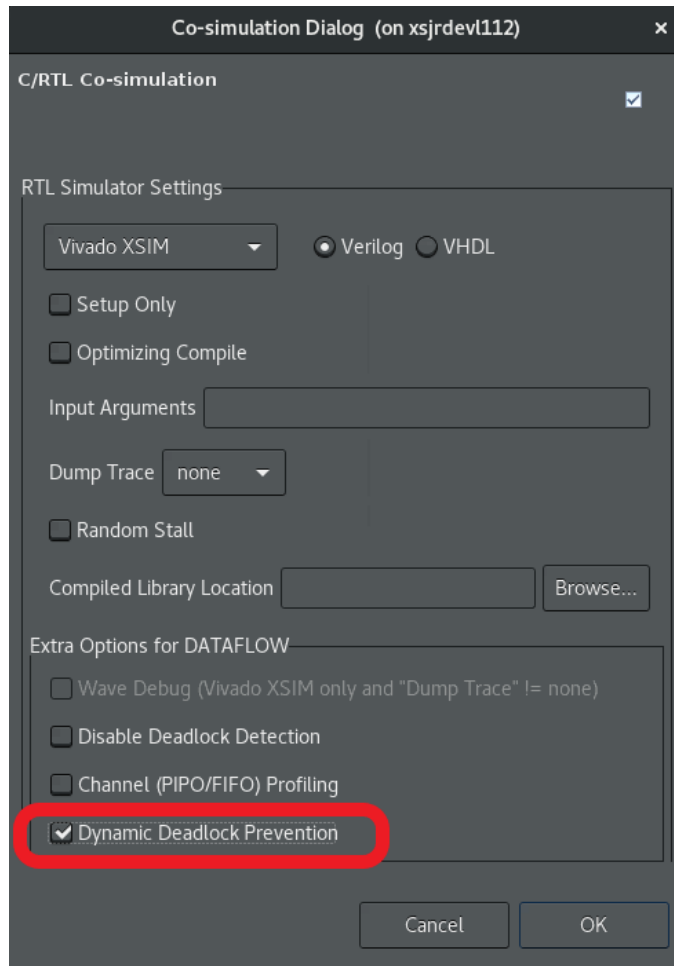
Global FIFO sizing

The screenshot displays the Vivado IDE interface. On the left, the Explorer pane shows a project structure with folders for 'proj', 'Includes', 'Source', 'Test Bench', and 'solution1'. The 'solution1' folder is expanded, showing subfolders for 'constraints', 'directives.tcl', 'script.tcl', 'impl', 'sim', and 'syn'. Below the Explorer, the Git Repositories pane offers options to add, clone, or create a repository. The main workspace shows a Dataflow graph with two main processing blocks, 'proc_1_U0' and 'proc_2_U0', each containing sub-processes and data channels. The bottom pane displays a table of FIFO properties for various data channels.

Process	Channel	Name	Cosim Category	Cosim Read Block Time	Cosim Write Block Time	Cosim Max Depth	Depth	Type	Sub-Type	BitWidth	Producer	Consumer	Cosim Distribution Graph
		data_channel1	read_block	10.00%	0.00%	10	40	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
		data_channel2	read_block	20.00%	0.00%	1	40	FIFO	Stream	32	proc_1_U0	proc_2_U0	N/A
		data_channel3	none	0.00%	0.00%	10	40	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
		data_channel4	read_block	10.00%	0.00%	1	40	FIFO	Stream	32	proc_1_1_U0	proc_1_2_U0	N/A
		data_channel5	read_block	20.00%	0.00%	10	40	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A
		data_channel6	read_block	56.92%	0.00%	1	40	FIFO	Stream	32	proc_2_1_U0	proc_2_2_U0	N/A

Second Lab – FIFO Sizing for Performance and avoiding Deadlocks

Automated FIFO sizing



Process	Channel	FIFO Sizing
Name	Suggest Depth	Type
data_channel1	11	Stream
data_channel2	2	Stream
data_channel1	11	Stream
data_channel2	2	Stream
data_channel1	11	Stream
data_channel2	2	Stream